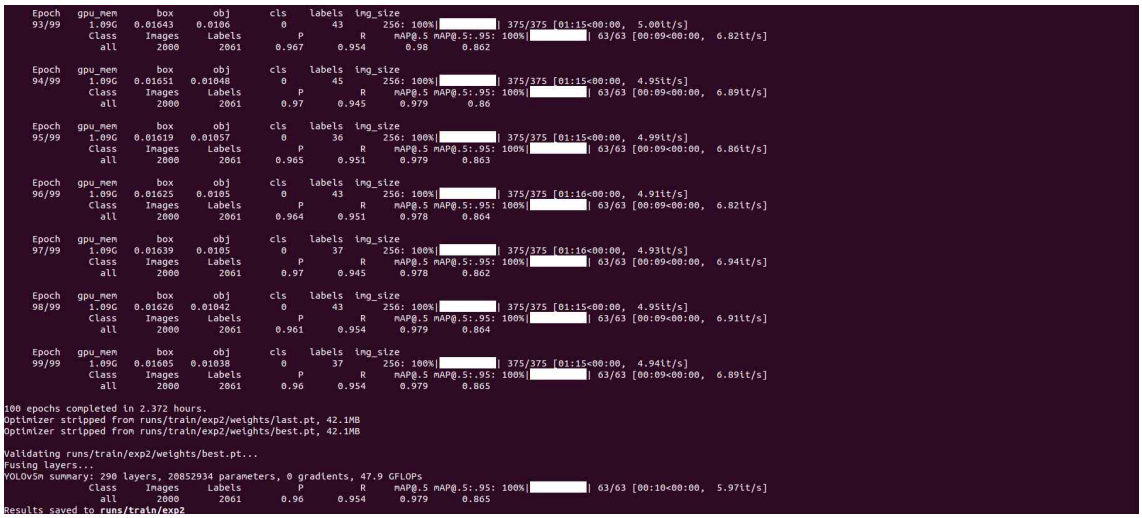
[CV Homework] Ozgenel(METU): Concrete Crack Images

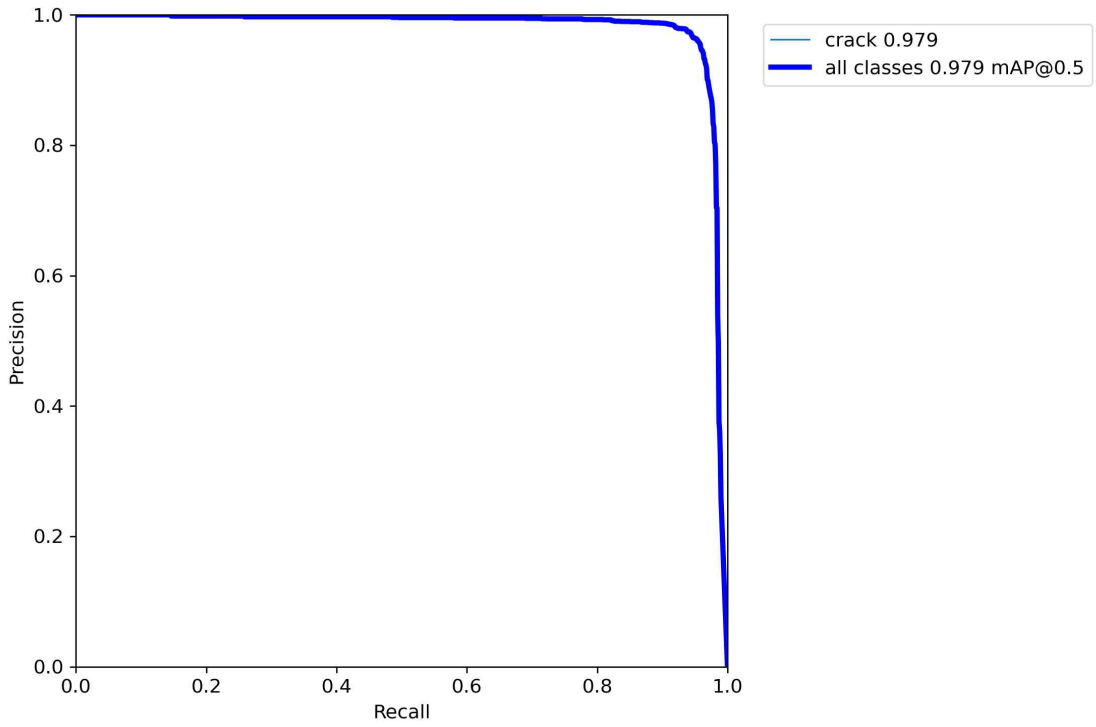
학습 전략:

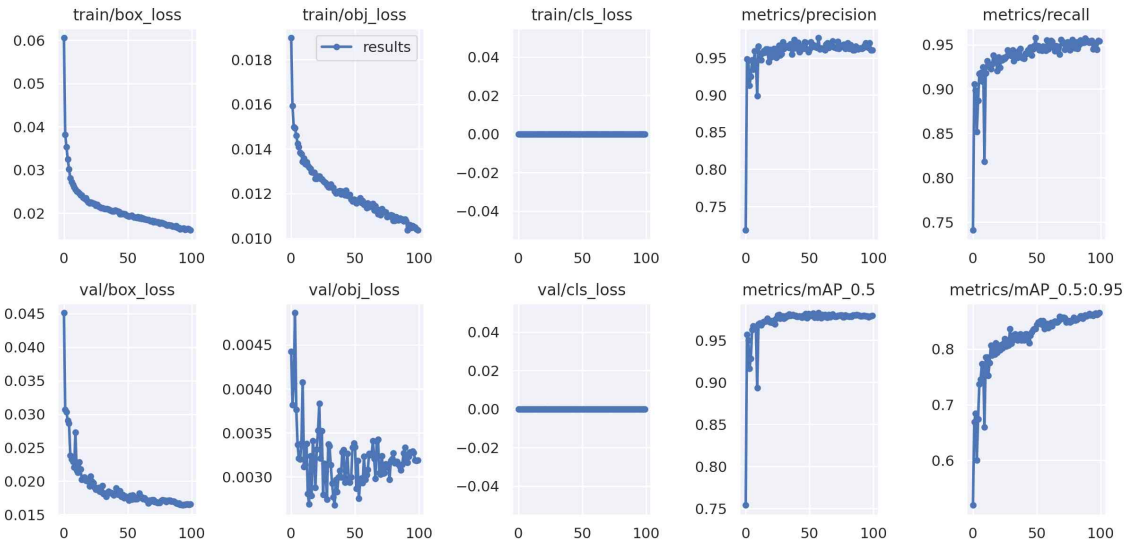
yolo에서는 모델의 성능(정확도)을 Mean average precision(mAP)를 통해 확인합니다. mAP이 높을수록 정확하고, 작을수록 부정확합니다. 좋은 성능의 모델을 만들기 위해서 먼저 yolov5의 모델을 살펴 보았습니다. 사용 가능한 모델은 총 5가지가 있습니다. yolov5l, yolov5m, yolov5n, yolov5s 그리고 yolov5x가 있었습니다. 수업 시간에 단시간에 결과를 확인해보기 위해서 가장 가벼운 모델인 yolov5s를 사용했었습니다. 그래서 이번 과제때는 조금 무거운 모델을 사용하면 좋은 결과를 낼거라고 생각했습니다. 그 결과 모델은 yolovx와 yolovl두가지를 선택했습니다. 그리고 저희가 테스트할 이미지들을 보았는데 227픽셀임을 확인했습니다. image size를 228로 지정해주었습니다. batch-size는 default가 32이지만 batch-size가 작을수록 더 높은 성능을 보인다고 생각을 해서 16으로 진행을 했습니다. 마지막으로 하이퍼 파리미터를 확인해보았습니다. 하이퍼 파라미터는 5가지가 있습니다. hyp.Objects365, hyp.VOC, hyp.scratch-high, hyp.scratch-med, hyp.scratch-low. 하이퍼 파라미터 안의 내용을 보니 hyp.objects365와 hyp.VOC는 다른 하이퍼 파라미터에 비해 좀 가벼운느낌을 받았습니다. 그리고 hyp.scratch-high, hyp.scratch-med, hyp.scratch-low를 비교해본 결과 큰 차이가 없다는 것을 확인해 hyp.scratch-med으로 학습을 진행 했습니다.

결과:

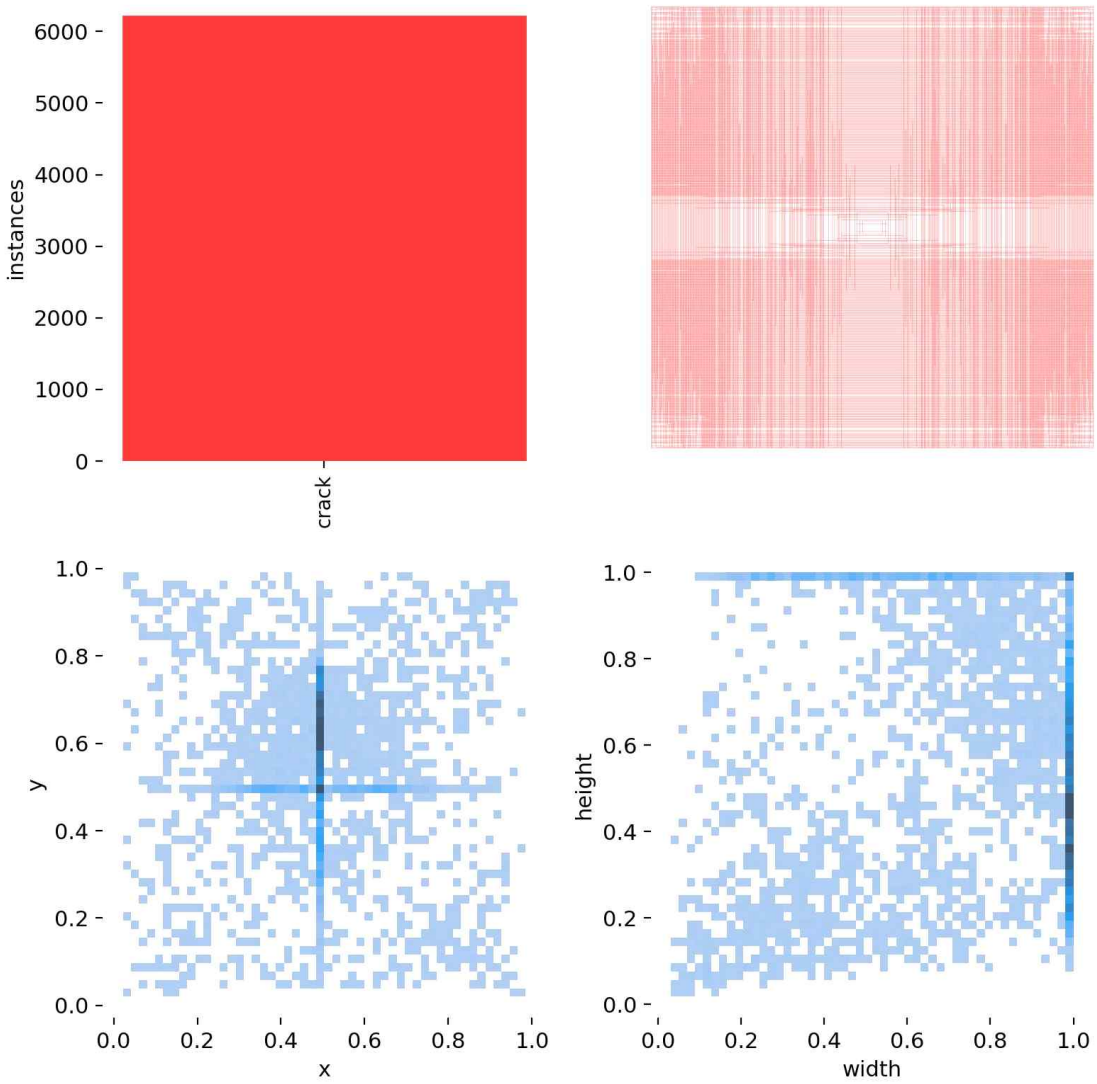


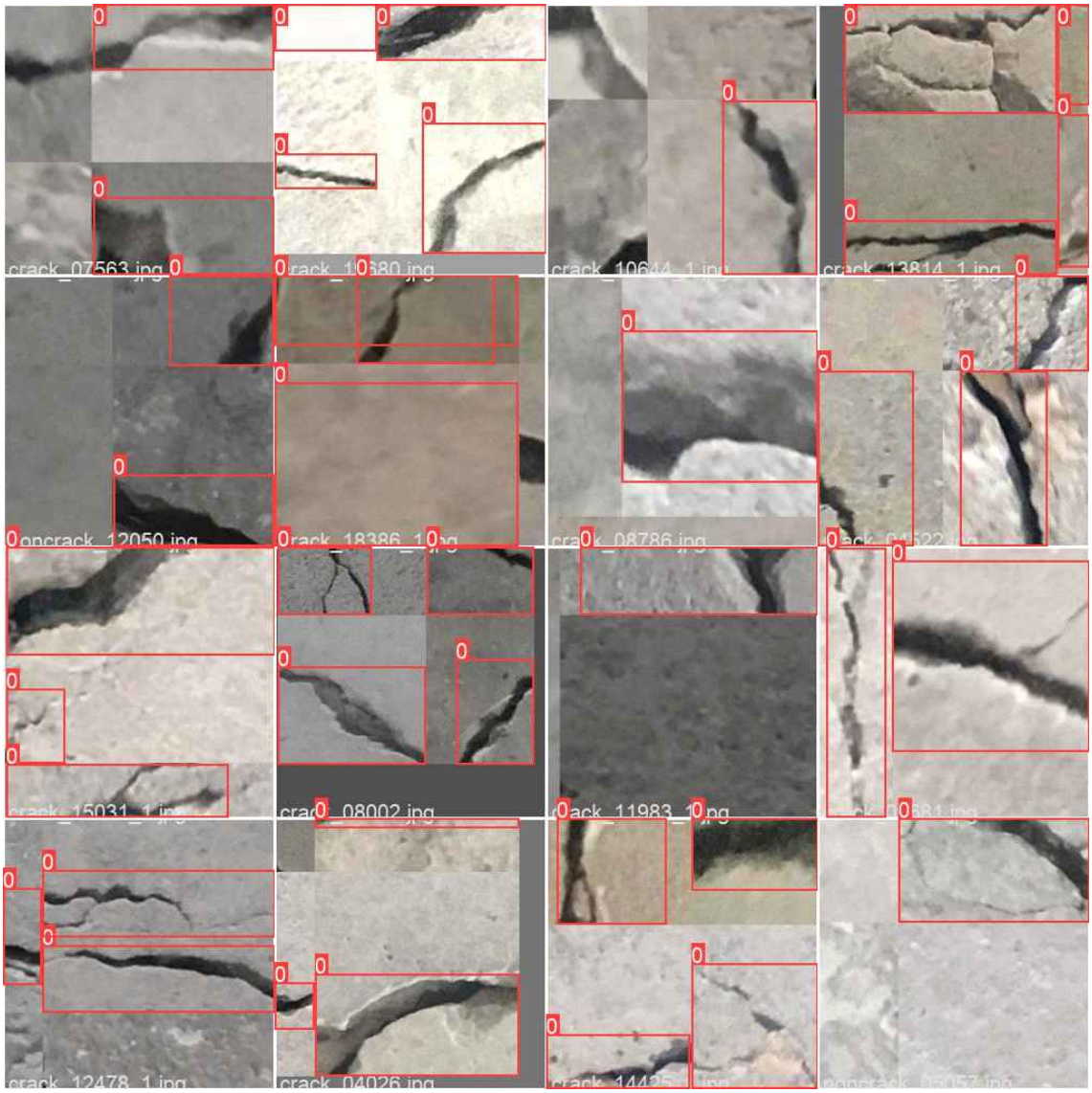
```
Epoch   gpu_mem       box       obj       cls    labels  img_size
89/99     1.09G   0.01658   0.01085         0        38       256: 100%|██████████| 375/375 [01:15<00:00,  4.99it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:08<00:00,  7.00it/s]
            all       2000       2061      0.963      0.955      0.979      0.859

Epoch   gpu_mem       box       obj       cls    labels  img_size
90/99     1.09G   0.01665   0.01078         0        47       256: 100%|██████████| 375/375 [01:15<00:00,  5.00it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.79it/s]
            all       2000       2061      0.963      0.953      0.979      0.861

Epoch   gpu_mem       box       obj       cls    labels  img_size
91/99     1.09G   0.01629   0.01037         0        40       256: 100%|██████████| 375/375 [01:14<00:00,  5.05it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.90it/s]
            all       2000       2061      0.963      0.952      0.979      0.863

Epoch   gpu_mem       box       obj       cls    labels  img_size
92/99     1.09G   0.01636   0.01056         0        48       256: 100%|██████████| 375/375 [01:15<00:00,  4.95it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.83it/s]
            all       2000       2061      0.964      0.954      0.979      0.861

Epoch   gpu_mem       box       obj       cls    labels  img_size
93/99     1.09G   0.01643    0.0106         0        43       256: 100%|██████████| 375/375 [01:15<00:00,  5.00it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.82it/s]
            all       2000       2061      0.967      0.954       0.98      0.862

Epoch   gpu_mem       box       obj       cls    labels  img_size
94/99     1.09G   0.01651   0.01048         0        45       256: 100%|██████████| 375/375 [01:15<00:00,  4.95it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.89it/s]
            all       2000       2061       0.97      0.945      0.979       0.86

Epoch   gpu_mem       box       obj       cls    labels  img_size
95/99     1.09G   0.01619   0.01057         0        36       256: 100%|██████████| 375/375 [01:15<00:00,  4.99it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.86it/s]
            all       2000       2061      0.965      0.951      0.979      0.863

Epoch   gpu_mem       box       obj       cls    labels  img_size
96/99     1.09G   0.01625    0.0105         0        43       256: 100%|██████████| 375/375 [01:16<00:00,  4.91it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.82it/s]
            all       2000       2061      0.964      0.951      0.978      0.864

Epoch   gpu_mem       box       obj       cls    labels  img_size
97/99     1.09G   0.01639    0.0105         0        37       256: 100%|██████████| 375/375 [01:16<00:00,  4.93it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.94it/s]
            all       2000       2061       0.97      0.945      0.978      0.862

Epoch   gpu_mem       box       obj       cls    labels  img_size
98/99     1.09G   0.01626   0.01042         0        43       256: 100%|██████████| 375/375 [01:15<00:00,  4.95it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.91it/s]
            all       2000       2061      0.961      0.954      0.979      0.864

Epoch   gpu_mem       box       obj       cls    labels  img_size
99/99     1.09G   0.01605   0.01038         0        37       256: 100%|██████████| 375/375 [01:15<00:00,  4.94it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:09<00:00,  6.89it/s]
            all       2000       2061       0.96      0.954      0.979      0.865
```

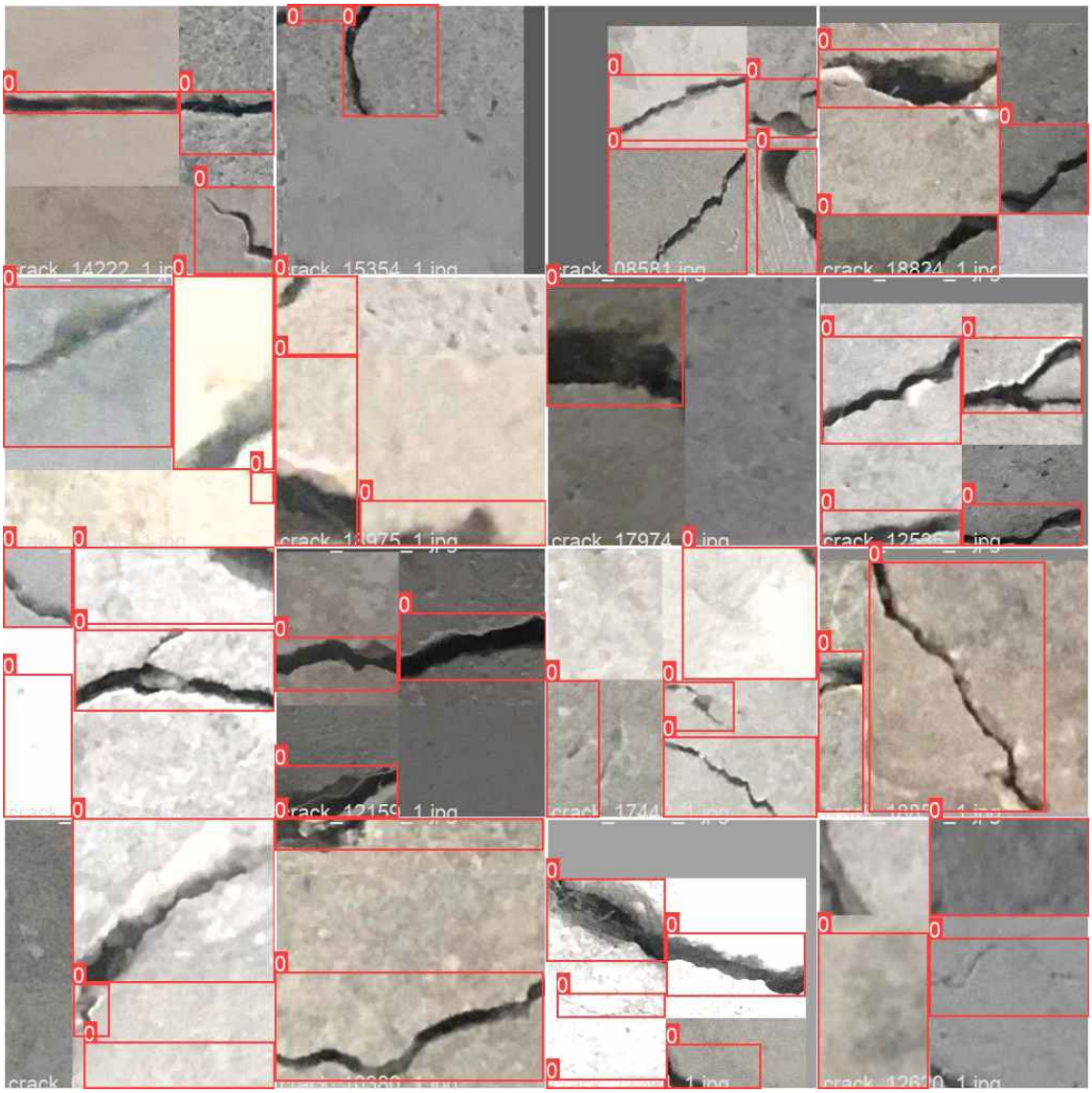


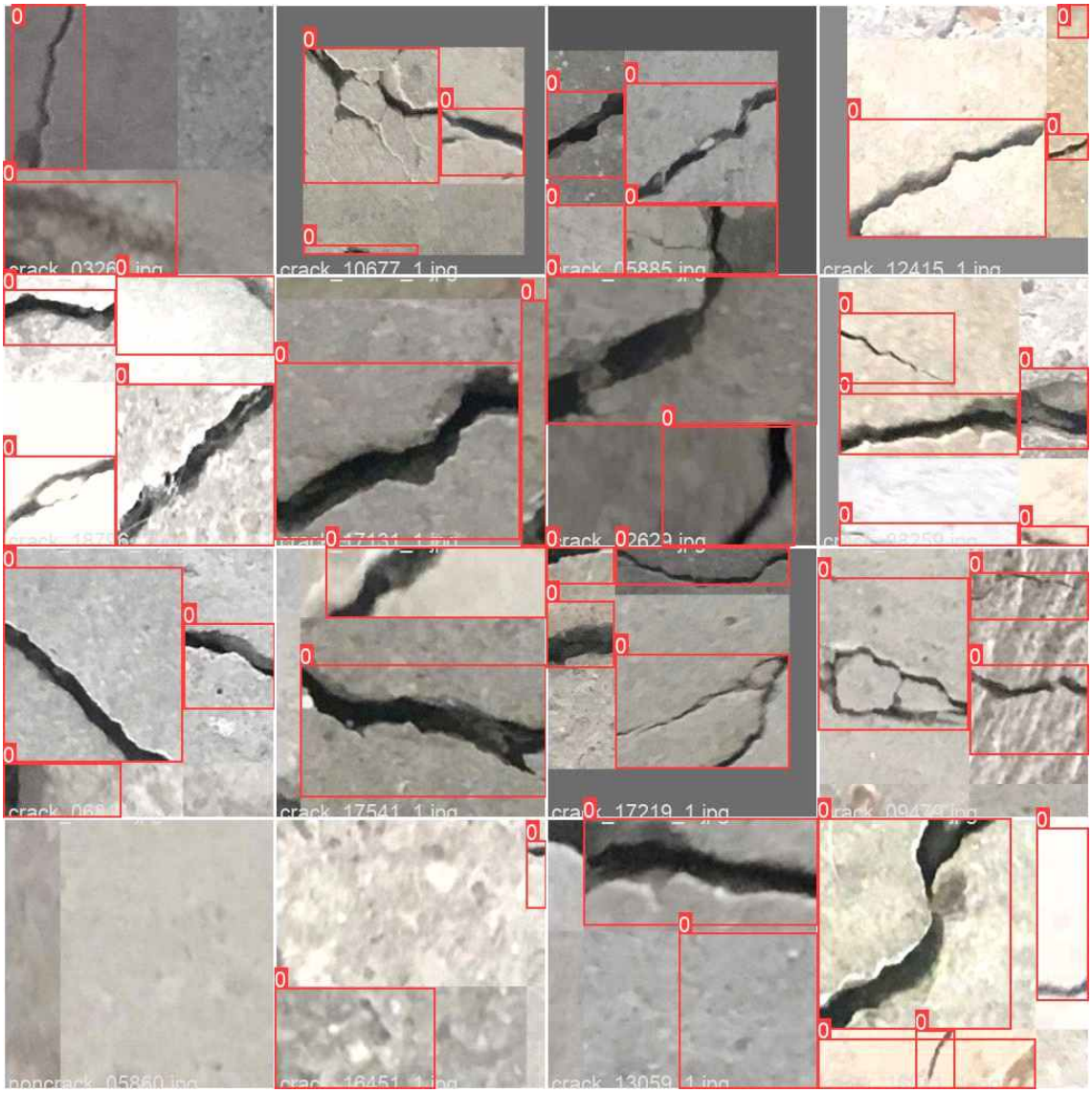

train/box_loss
train/obj_loss
results
train/cls_loss
metrics/precision
metrics/recall
val/box_loss
val/obj_loss
val/cls_loss
metrics/mAP_0.5
metrics/mAP_0.5:0.95

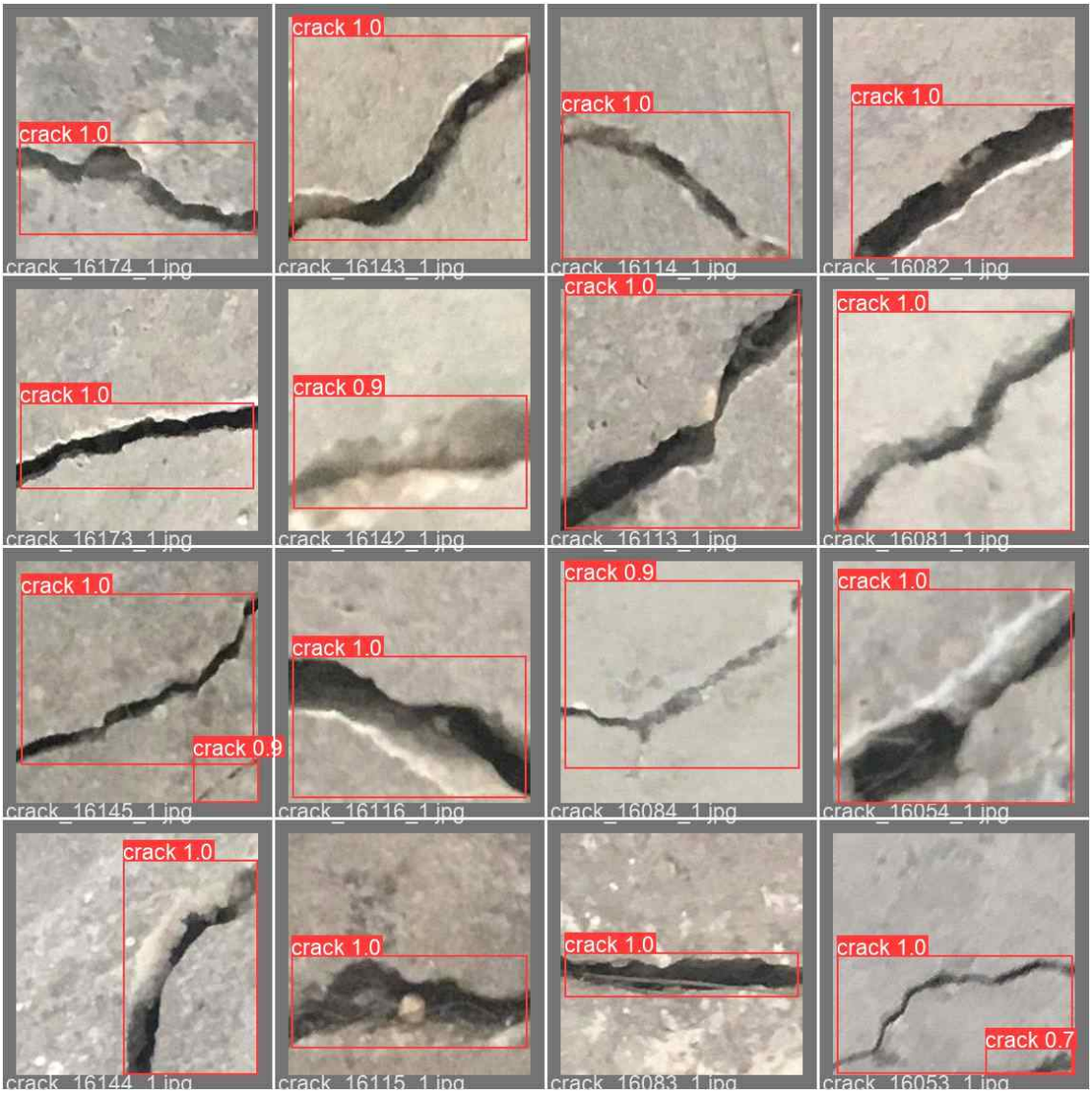
개선할수 있는 방향:

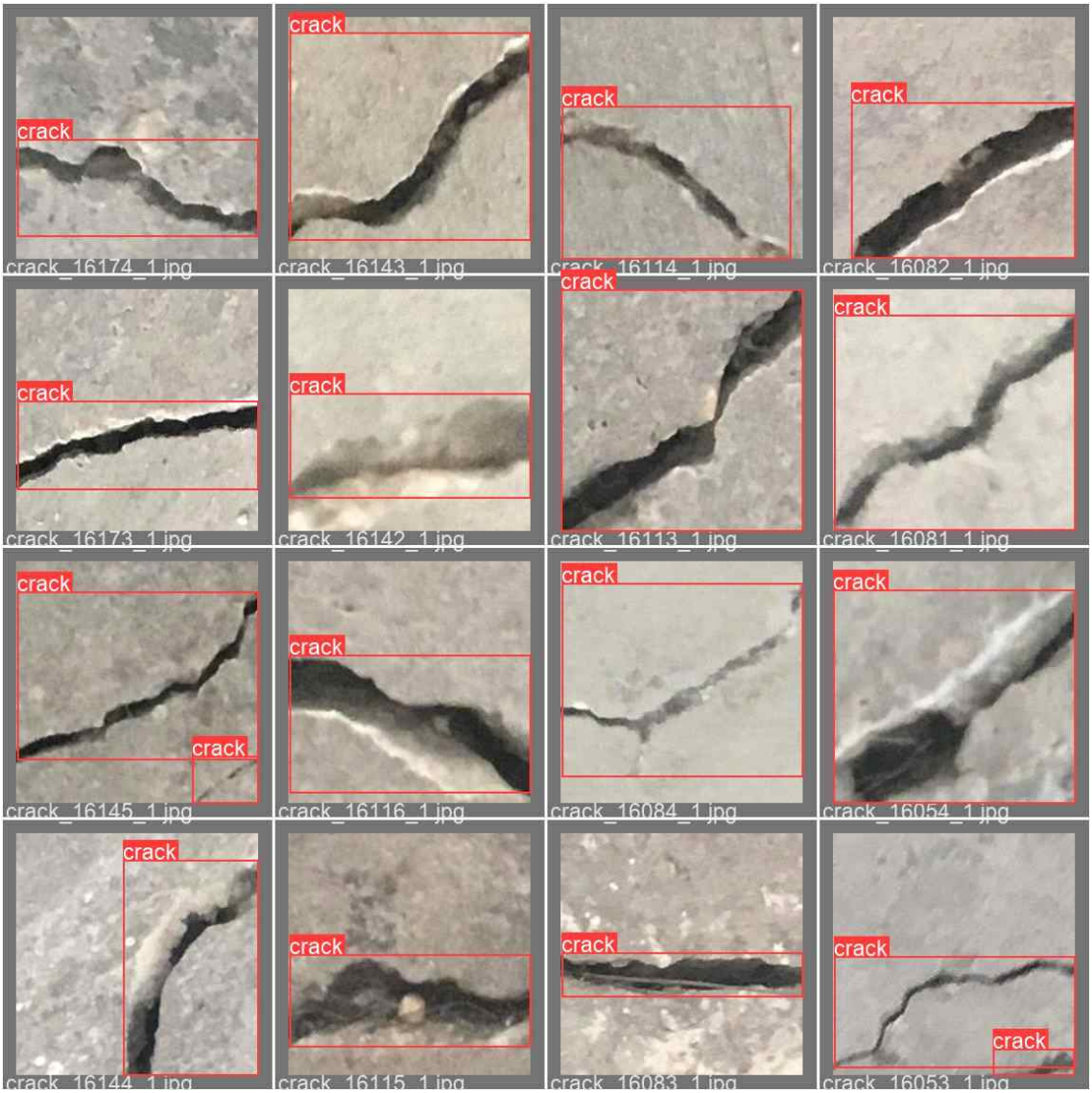

instances
6000
5000
4000
3000
2000
1000
0
crack
y
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
x
height
width

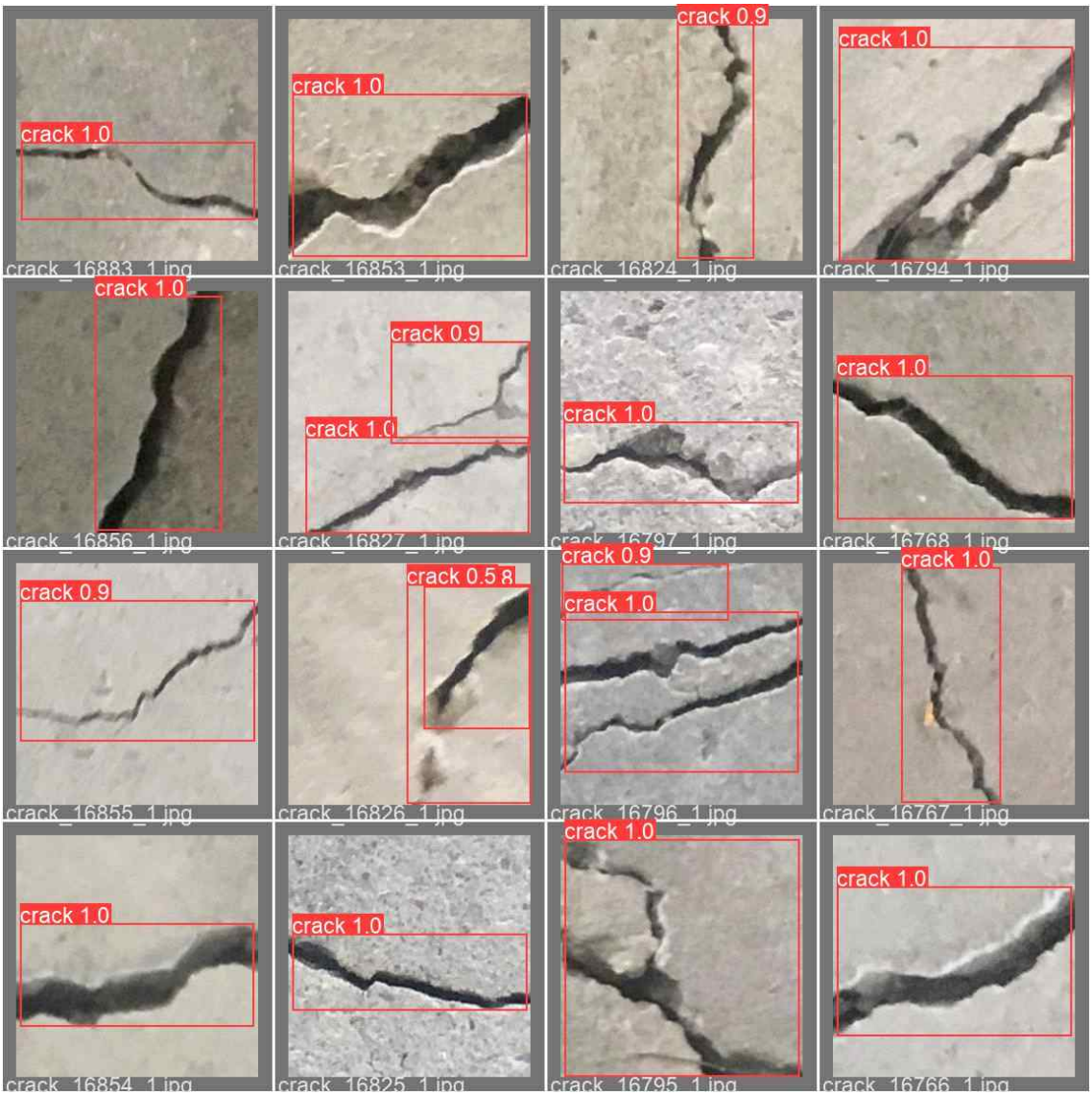


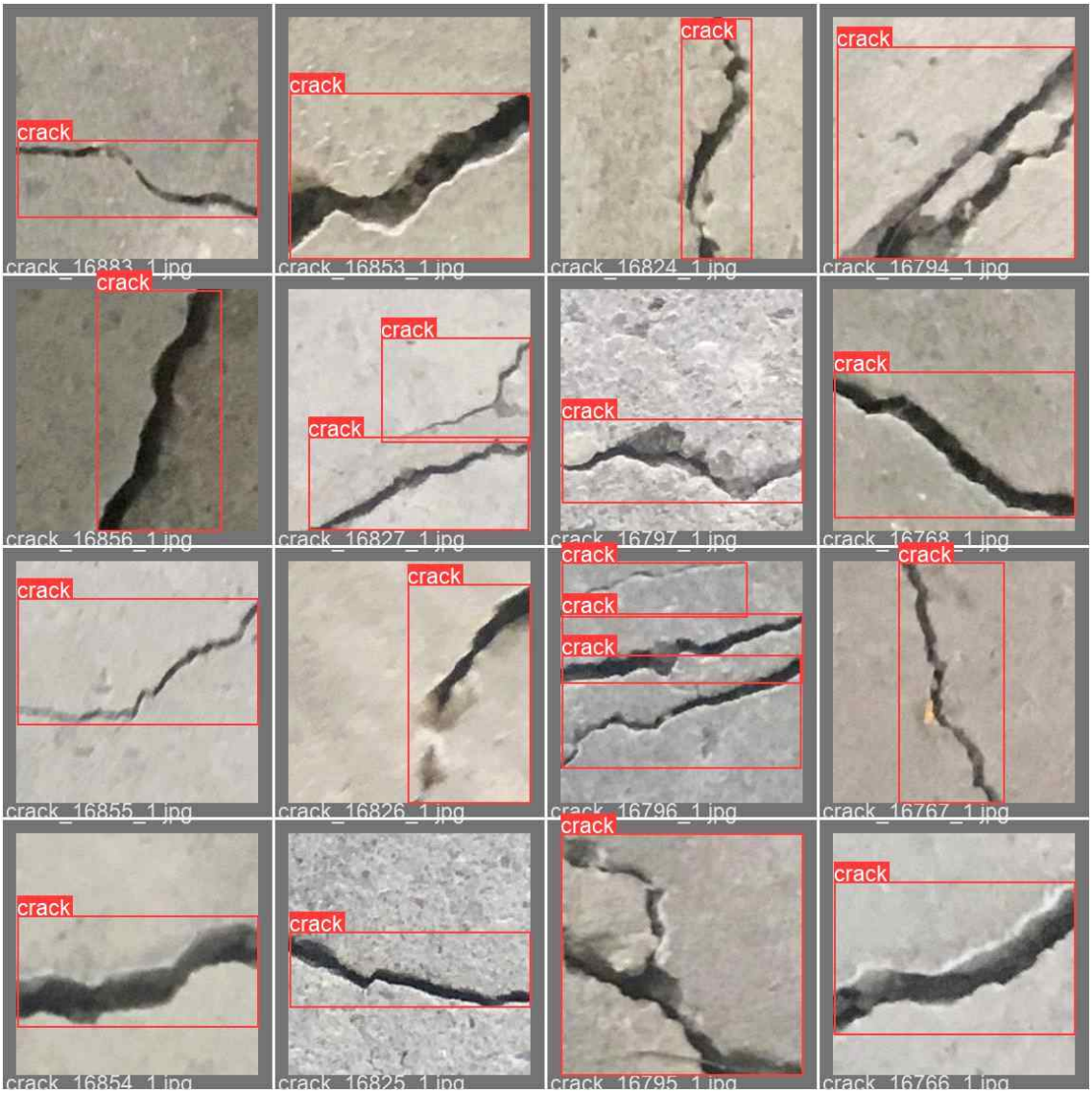


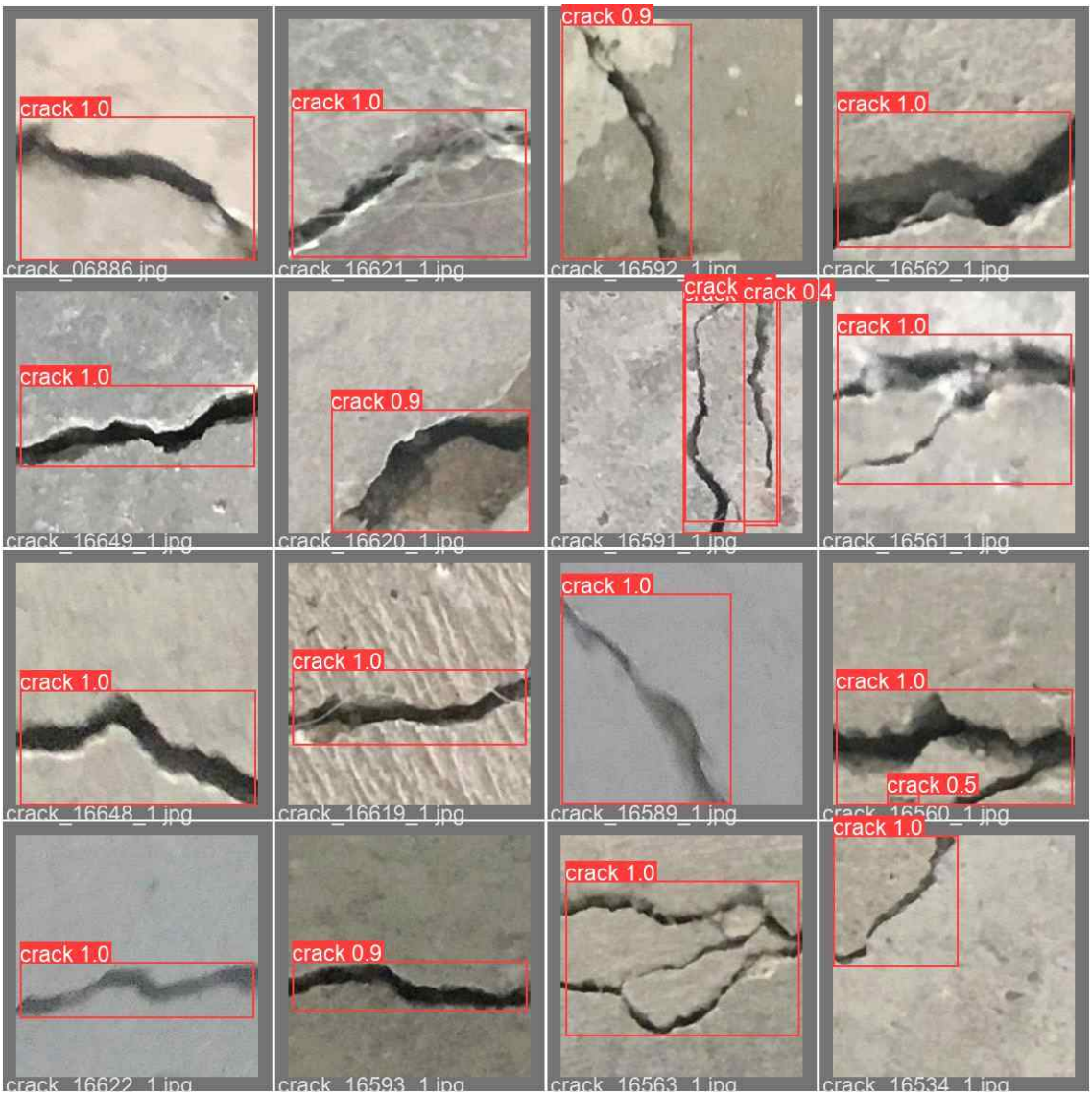

crack_10677_1.jpg
crack_12415_1.jpg
crack_17541_1.jpg
crack_17219_1.jpg
crack_13059_1.jpg

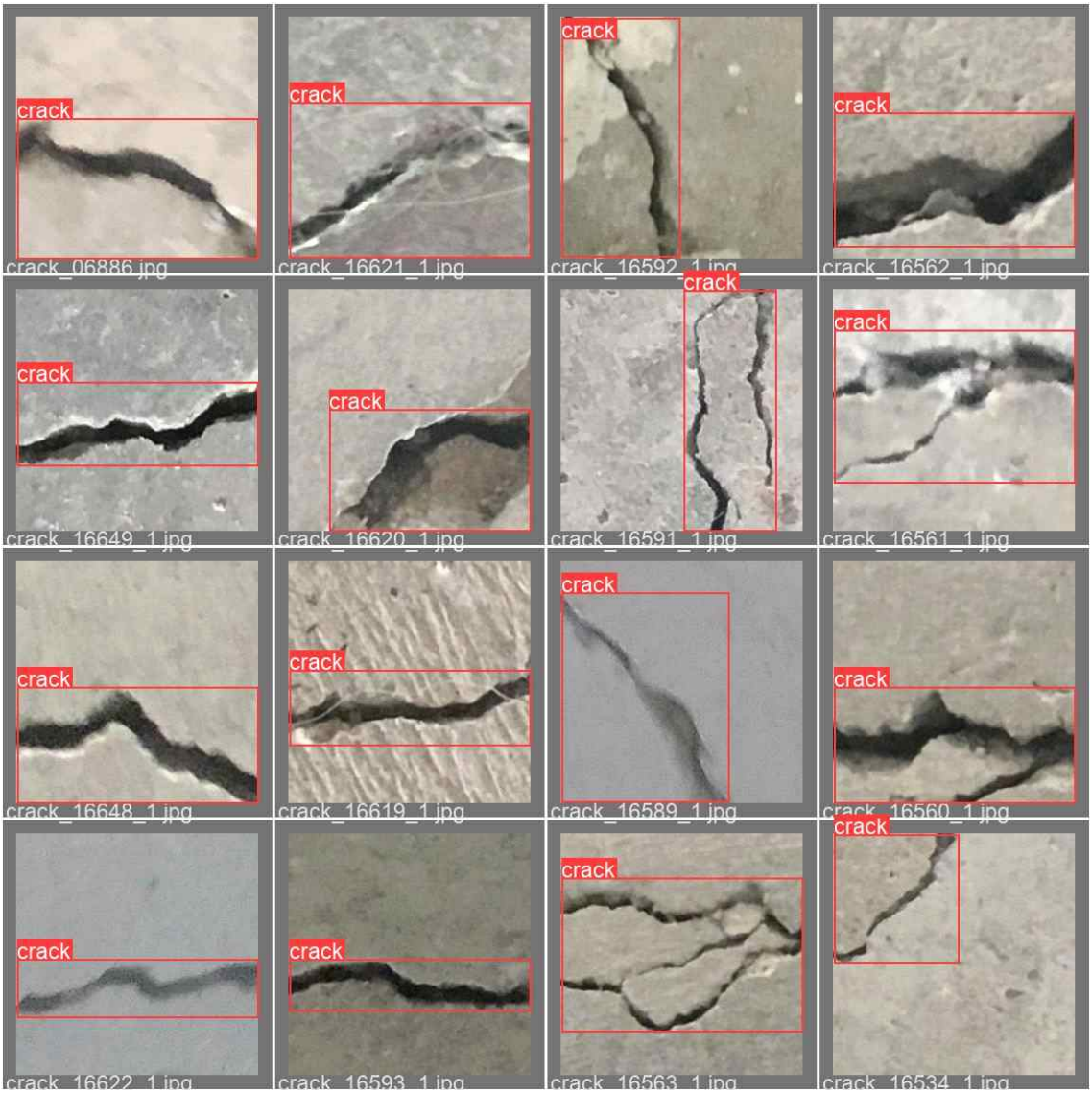

crack 1.0
crack_16174_1.jpg
crack 1.0
crack_16143_1.jpg
crack 1.0
crack_16114_1.jpg
crack 1.0
crack_16082_1.jpg
crack 1.0
crack_16173_1.jpg
crack 0.9
crack_16142_1.jpg
crack 1.0
crack_16113_1.jpg
crack 1.0
crack_16081_1.jpg
crack 1.0
crack 0.9
crack_16145_1.jpg
crack 1.0
crack_16116_1.jpg
crack 0.9
crack_16084_1.jpg
crack 1.0
crack_16054_1.jpg
crack 1.0
crack_16144_1.jpg
crack 1.0
crack_16115_1.jpg
crack 1.0
crack_16083_1.jpg
crack 1.0
crack 0.7
crack_16053_1.jpg


crack
crack_16174_1.jpg
crack
crack_16143_1.jpg
crack
crack_16114_1.jpg
crack
crack_16082_1.jpg
crack
crack_16173_1.jpg
crack
crack_16142_1.jpg
crack
crack_16113_1.jpg
crack
crack_16081_1.jpg
crack
crack
crack_16145_1.jpg
crack
crack_16116_1.jpg
crack
crack_16084_1.jpg
crack
crack_16054_1.jpg
crack
crack_16144_1.jpg
crack
crack_16115_1.jpg
crack
crack_16083_1.jpg
crack
crack
crack_16053_1.jpg


crack 1.0
crack_16883_1.jpg
crack 1.0
crack_16853_1.jpg
crack 0.9
crack_16824_1.jpg
crack 1.0
crack_16794_1.jpg
crack 1.0
crack_16856_1.jpg
crack 0.9
crack 1.0
crack_16827_1.jpg
crack 1.0
crack_16797_1.jpg
crack 1.0
crack_16768_1.jpg
crack 0.9
crack_16855_1.jpg
crack 0.58
crack_16826_1.jpg
crack 0.9
crack 1.0
crack_16796_1.jpg
crack 1.0
crack_16767_1.jpg
crack 1.0
crack_16854_1.jpg
crack 1.0
crack_16825_1.jpg
crack 1.0
crack_16795_1.jpg
crack 1.0
crack_16766_1.jpg


crack
crack_16883_1.jpg
crack
crack_16853_1.jpg
crack
crack_16824_1.jpg
crack
crack_16794_1.jpg
crack
crack_16856_1.jpg
crack
crack
crack_16827_1.jpg
crack
crack_16797_1.jpg
crack
crack_16768_1.jpg
crack
crack_16855_1.jpg
crack
crack_16826_1.jpg
crack
crack
crack
crack_16796_1.jpg
crack
crack_16767_1.jpg
crack
crack_16854_1.jpg
crack
crack_16825_1.jpg
crack
crack_16795_1.jpg
crack
crack_16766_1.jpg


crack 1.0
crack_06886.jpg
crack 1.0
crack_16621_1.jpg
crack 0.9
crack_16592_1.jpg
crack 1.0
crack_16562_1.jpg
crack 1.0
crack_16649_1.jpg
crack 0.9
crack_16620_1.jpg
crack 0.4
crack_16591_1.jpg
crack 1.0
crack_16561_1.jpg
crack 1.0
crack_16648_1.jpg
crack 1.0
crack_16619_1.jpg
crack 1.0
crack_16589_1.jpg
crack 1.0
crack 0.5
crack_16560_1.jpg
crack 1.0
crack_16622_1.jpg
crack 0.9
crack_16593_1.jpg
crack 1.0
crack_16563_1.jpg
crack 1.0
crack_16534_1.jpg



많은 시도를 해보았지만 생각보다 높은 정확도가 나오지 않았습니다. epoch의 값을 좀더 늘리고 사용한 hyp.scratch-med의 값을 조금 변경 했더라면 더 좋은 결과가 나올수 있을거 같다는 생각이 들었습니다.